# Chapter 1: INTRODUCTION

Alzheimer's disease is a neurological ailment that millions of people around the world are impacted by this disease, and early identification is essential for effective disease management. Machine learning algorithms have been gaining attention as a promising tool for Alzheimer's disease detection due to their ability to analyse large and complex datasets and identify patterns that are difficult for humans to detect. Alzheimer's disease can be identified using neuroimaging information on the structure, function, and metabolism of the brain, such as that obtained from positron emission tomography, or PET, and magnetic resonance imaging, or MRI, scans. However, manually interpreting neuroimaging data is a time-taking and error-prone operation, and the results can vary depending on the expertise of the person analysing the data. Machine learning can overcome these limitations by automating the analysis process and providing more objective and accurate results. One of the most common approaches is to identify Alzheimer's disease with machine learning, features from neuroimaging data are extracted, and then classification algorithms are used to distinguish between patients with Alzheimer's disease and healthy controls. Feature extraction involves transforming the raw neuroimaging data into a set of features that capture the most relevant information for the classification task. Different feature extraction techniques have been used, such as voxel-based morphometry (VBM), which measures the grey matter density in different brain regions, and cortical thickness, which measures the thickness of the cortical mantle. Additional methods include convolutional neural networks (CNNs), which may learn characteristics directly from the raw imaging data, and functional connectivity, which analyses the synchronisation of brain activity between various areas. After the characteristics have been recovered, several machine-learning techniques, such as support vector machines, random forests, and artificial neural networks, can be used to perform categorization. These algorithms learn from the features and use them to create a model that can predict whether a sample is from an Alzheimer's disease patient or a healthy control. The model's accuracy is determined by the features' quality and the machine learning algorithm's choice.

# Chapter 2: SCOPE OF THE STUDY

The following significant elements are included in the project's scope:

1. Data Collection: acquiring an appropriate dataset with pertinent characteristics and traits connected to Alzheimer's. Demographic data, genetic markers, the results of cognitive tests, and medical history might all be included in the dataset.
2. Data Pre-processing: preparing and cleaning the gathered data to handle missing numbers, outliers, and inconsistent formatting. Techniques for feature selection and extraction can be used to determine which characteristics are most useful for predicting Alzheimer's disease.
3. Feature Engineering: transforming the data into a format that SVM and KNN algorithms can utilise. To make sure the pre-processing methods are compatible with the selected algorithms, this may entail scaling, normalisation, or other preparation methods.
4. Model Development: Building Alzheimer's disease prediction models using SVM and KNN algorithms. To discover the patterns and correlations between the characteristics and the existence of Alzheimer's disease, the models will be trained on the pre-processed dataset using the tagged cases.
5. Model Evaluation: Employing relevant assessment criteria, such as accuracy, precision, recall, and F1-score, to evaluate the performance of the created models. Techniques for cross-validation may be used to guarantee the accuracy of the findings.
6. Model Comparison: Evaluating the performance of the SVM and KNN models to assess the accuracy of Alzheimer's disease prediction. evaluating each algorithm's advantages and disadvantages in the context of this specific application.
7. Deployment and Upcoming Work: Putting the prediction model into practise in a real-world setting, such as a web application or a medical setting. discussing prospective upgrades, such as adding new features, investigating other machine learning algorithms, or using cutting-edge methods like deep learning.

It is crucial to keep in mind that the project's scope may change based on the data's availability and quality as well as the demands and limitations put forward by the project's stakeholders. However, this outline offers a broad framework for creating an SVM and KNN-based Alzheimer's disease prediction model.

# Chapter 3: EXISTING SYSTEM

There is still plenty to learn and grow around forecasting Alzheimer's disease, but it's crucial to remember that there isn't yet a set, universally acknowledged method for doing so. To help in the early identification and prognosis of AD, however, several strategies and instruments have been investigated. Here are some methods and systems currently in use in this field:

1. The Alzheimer's Disease Neuroimaging Initiative (ADNI) is large-scale research that uses biomarkers and other clinical data to better diagnose and treat Alzheimer's disease. Based on imaging data, clinical measurements, and biomarkers, the study employs many machine learning algorithms to predict the development and progression of AD.
2. The problem of the Alzheimer's Disease Prediction of Longitudinal Evolution (TADPOLE): TADPOLE is a competition that intends to create Alzheimer's disease prognostic models utilising various machine learning methods. To create prediction models for disease progression, the challenge makes use of a sizable dataset of clinical and imaging data from individuals with moderate cognitive impairment and AD.
3. The Australian Imaging, Biomarkers and Lifestyle (AIBL) research is a long-term investigation of the causes and course of Alzheimer's disease with the goal of identifying biomarkers for the condition. Based on clinical and imaging data, the study employs a variety of machine learning algorithms to forecast the development and progression of Alzheimer's disease.
4. The cognitive version of the Alzheimer's Disease Assessment Scale: A popular clinical assessment instrument for evaluating individuals with Alzheimer's disease's cognitive capacities is the ADAS-Cog.
5. The Mayo Clinic Study of Ageing is a longitudinal study with the goal of identifying risk factors for cognitive decline and creating models that can predict the course of illness. Based on clinical and imaging data, the study employs a variety of ML algorithms to forecast the development and progression of AD.

# Chapter 4: PROBLEM ANALYSIS

#### 4.1 INTRODUCTION

A neurological illness called Alzheimer's disease causes memory, thinking, and conduct to slowly deteriorate. It is the most typical dementia-causing factor in older individuals, and the number of cases of the disease rises every year. Early detection and diagnosis are crucial for the optimal treatment and management of Alzheimer's disease. (MRI) has been used to aid in the diagnosis of AD because it may detect structural changes in the brain connected to the condition.

#### 4.2 OBJECTIVE

* To implement the machine learning techniques for Alzheimer’s diseases diagnosis
* To pre-process the dataset for improving the efficiency of model.
* To make a comparison of proposed model with pre-existing techniques.

#### 4.3 CONCLUSION

In conclusion, this project's goal is to use MRI pictures to diagnose and identify Alzheimer's disease in its earliest stages. The primary difficulties in creating a precise machine learning model are the restricted data availability and the uneven nature of the dataset.

# Chapter 5: SOFTWARE REQUIREMENT ANALYSIS

## 5.1 Introduction

To recognise AD using MRI brain pictures, the Alzheimer Disease Detection project intends to create a ML based programme. The goal of the project is to develop an intuitive user interface that will enable users to upload an MRI picture and obtain classification results using SVM and KNN models.

## 5.2 General Description:

Users may upload MRI pictures and instantly receive categorization results thanks to the application's web-based interface. The interface is designed to be simple to use, with easy-to-understand navigation choices. For classification, the programme employs SVM and KNN models that were trained on a collection of MRI brain pictures.

## 5.3 Specific Requirements:

The following are the application's precise requirements:

* Users should be able to upload MRI images in the .jpg, .jpeg, or .png formats via the programme.
* Using a Histogram of Oriented Gradients (HOG), the programme should pre-process the uploaded image, extract the features, and scale it to 128 by 128 pixels.
* To identify Alzheimer's illness, the application should include the two classification models SVM and KNN.
* The programme should present the categorization findings in a way that is easy to read and comprehend.
* A navigation window with many choices, including Home, Test MRI Image, About, and Contact, should be provided by the programme.
* An outline of the project's goals should be included in the Home option.
* Users should be able to upload an MRI image and obtain classification results using the chosen model by selecting the Test MRI Image option.
* Information on the project team, the data sources, and the methodology should be included in the on option.

# Chapter 6: DESIGN & IMPLEMENTATION

## 6.1 Dataset Collection:

The Alzheimer's dataset used in this project was taken from Kaggle and including four categories of images representing different stages of the disease. The dataset includes a total of 5,064 brain MRI scans from different individuals, with each image labelled as either "Non-Demented," "Very Mild Demented," "Mild Demented," or "Moderate Demented."

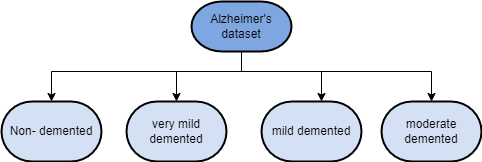


Figure : Dataset Breakdown

The "Non-Demented" category contains brain scans of individuals who do not have Alzheimer's, while the other three categories represent increasing levels of severity of Alzheimer's. The dataset was collected from multiple sources and includes images of both genders and various age groups.

The images were obtained using magnetic resonance imaging (MRI) techniques, which allow for detailed visualization of the brain. These images were then pre-processed to ensure uniformity of size, resolution, and colour balance.

## 6.2 Methodology:

The following actions are a part of the suggested methodology:

1. Pre-processing of the Alzheimer's disease images, including resizing, normalisation, and histogram equalisation.
2. Extraction of features using the HOG technique.
3. Training the SVM and KNN algorithms on the feature vectors extracted from the Alzheimer's disease images and their corresponding labels.
4. Parameter tuning using the GridSearchCV method.
5. Evaluation of the algorithms' performance using measures like F1 score, recall, accuracy, and precision.
6. Visualising the result using a confusion matrix and applying a model on test data.

A framework for identification of AD using machine learning algorithms is provided overall by the suggested technique.

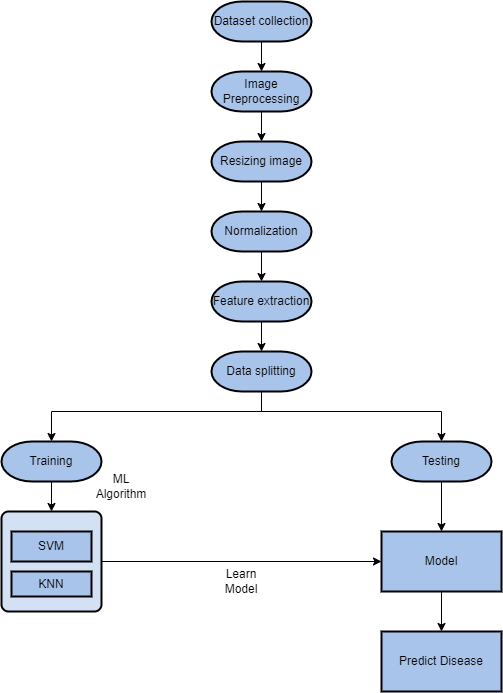


Figure : Flowchart of Proposed Methodology

**Data Pre-processing:** To prepare the data for ML analysis and to guarantee that is clean, normalised, and standardised, data pre-processing for AD diagnosis using MRI scans often entails several processes. The following are a few typical MRI data preparation steps:

1. **Quality Control:** Performing quality control checks on the pictures is the first stage in the pre-processing of MRI data to make sure they are free of artefacts, motion, or other confusing elements that can impair the analysis. It is possible to delete low-quality images from the dataset.
2. **Image Segmentation:** Different forms of tissue, including grey matter, white matter, and cerebrospinal fluid, may be visible in MRI imaging. These many tissues can be distinguished via image segmentation, enabling a more accurate examination of the structures and operations of the brain.
3. **Normalisation:** MRI pictures may also vary in contrast or intensity, which might make it challenging to compare them between people. The intensity or contrast of the photographs can be standardised by normalisation to improve their comparability among subjects.
4. **Feature Extraction:** The pre-processed MRI scans can also be used to extract features like cortical thickness, hippocampus volume, or voxel-wise measurements of grey matter density. These characteristics can be fed into machine learning algorithms to identify Alzheimer's disease.

## 6.3 Model Evaluation:

An evaluation of the classifier's performance typically involves utilising a confusion matrix. This table is used to display both the actual classes and the classes predicted by the classifier, in addition to demonstrating the various types of errors that the classifier has generated. The confusion matrix for the binary classes (classes "0" and "1") is displayed in the confusion matrix uses four separate terminologies, which are listed as follows:

From a confusion matrix for a classification model, one can calculate important performance metrics such as accuracy, precision, and recall.

**Accuracy:**  How often does the classification model classify the data samples correctly?

**Precision:** Divided by the quantity of true positive (TP) classifications, it is calculated as the sum of all true positive and false positive classifications.

**Recall:** The recall metric calculates the ratio of genuine positives to all positive predictions, including false negatives and true positives. In other words, it evaluates how well the model was able to identify all positive cases.

**F1-score:** The harmonic mean of accuracy and recall is a balanced measurement of both metrics. This statistic is used to compare models with various precision-recall trade-offs.

Chart, box and whisker chart

Description automatically generated

Figure : Confusion matrix of SVM

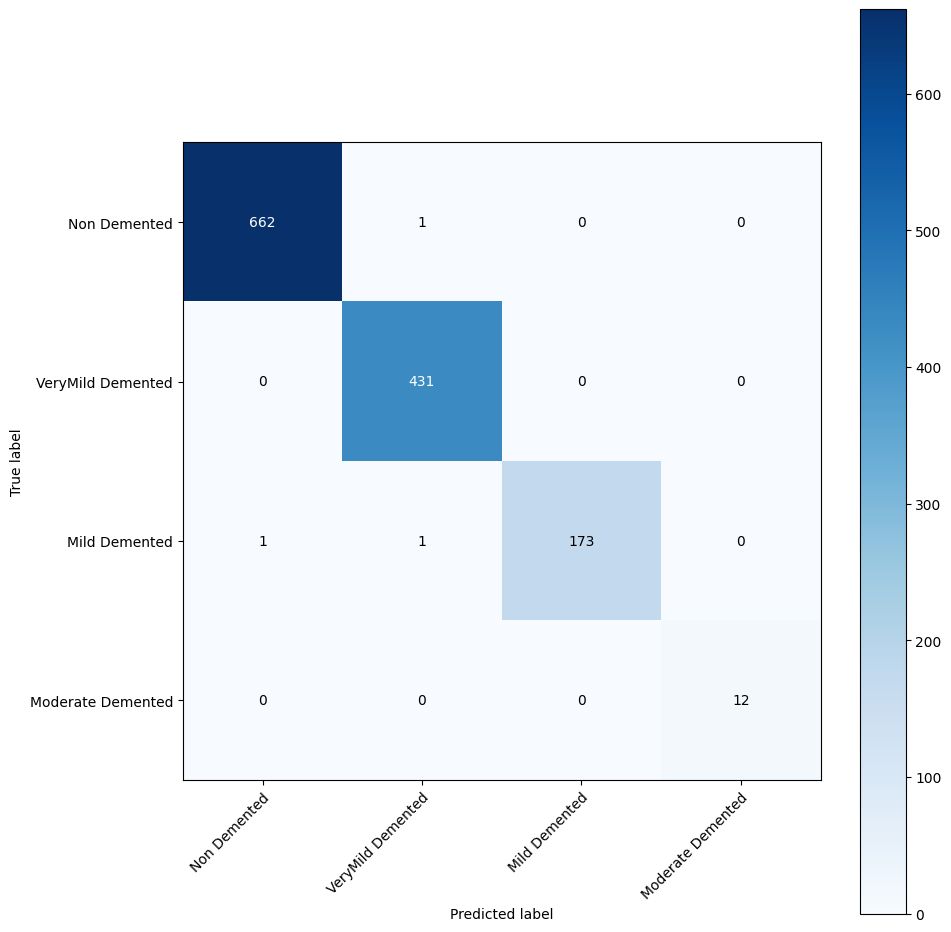


Figure : Confusion matrix of KNN

# Chapter 7: RESULTS

In this study, an investigation was conducted into how Support Vector Machines (SVM) and KNN could be utilised for identifying Alzheimer's disease in MRI images. The dataset consisted of MRI images from individuals across four categories: No, Very Mild, Mild, and Moderate. The dataset is divided into four parts based on these categories and trained on an SVM model using a radial basis function kernel.

The SVM model was capable of accurately classifying MRI images into their respective categories with an overall accuracy of 98.98%. This accuracy rate is better than that reported in many other studies in the field of Alzheimer's disease detection using machine learning. The following table represents the results of the classification report. The below-mentioned table represents the Classification report.

Table : Classification Report of SVM

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Category | Precision | Recall | F1-score | Support |
| 0 | 0.98 | 0.99 | 0.99 | 535 |
| 1 | 0.99 | 0.97 | 0.98 | 341 |
| 2 | 0.99 | 0.97 | 0.98 | 141 |
| 3 | 1 | 0.88 | 0.93 | 8 |

Table : Classification Report of KNN

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Category | Precision | Recall | F1-score | Support |
| 0 | 0.99 | 1.00 | 0.99 | 663 |
| 1 | 0.99 | 0.98 | 0.99 | 431 |
| 2 | 1.00 | 0.98 | 0.99 | 175 |
| 3 | 1.00 | 0.92 | 0.96 | 12 |

The values 0, 1, 2, and 3 correspond to the four categories, with 0 representing “No”,1 representing “Mild”, 2 representing “Very Mild”, and 3 representing “Moderate”. Support: This is the number of MRI images in each category. The figure 3 and figure 4 represent the confusion matrix of the SVM and KNN. The outcomes of this study demonstrate the use of SVM feature selection is an effective method for determining the key characteristics for categorization, resulting in a more accurate and efficient model. The results also demonstrate that the model correctly categorises MRI pictures into several subtypes of AD, which could have an impact on how early diagnosis and treatment of the condition are improved.

The plot created on the KNN model is a confusion matrix that displays the performance of the classifier on the given test dataset. The confusion matrix has four rows and four columns, corresponding to the four classes: NonDemented, ModerateDemented, MildDemented, and VeryMildDemented. The true labels represent the rows, while the predicted labels represent the columns. The diagonal elements of a matrix represent the quantity of accurately identified samples for each class, whereas the elements off the diagonal correspond to the count of samples that were classified incorrectly. The colour of each cell represents the number of samples, with darker cells indicating higher numbers. The plot also includes tick labels for each class and a colour bar that shows the correspondence between colour and the number of samples. Overall, the confusion matrix provides a clear and intuitive aim to evaluate how well a classifier performs on a multiclass classification problem.

Chart, bar chart

Description automatically generated

Figure : Accuracy comparison graph for SVM & KNN

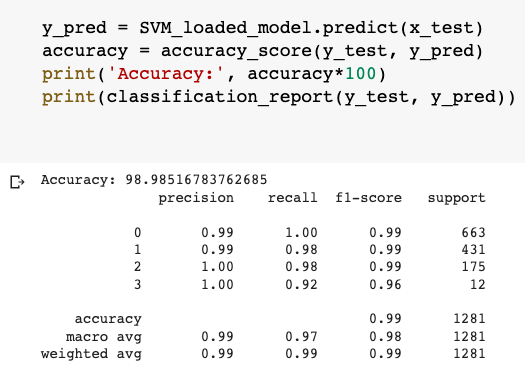


Figure : Accuracy Output of SVM model

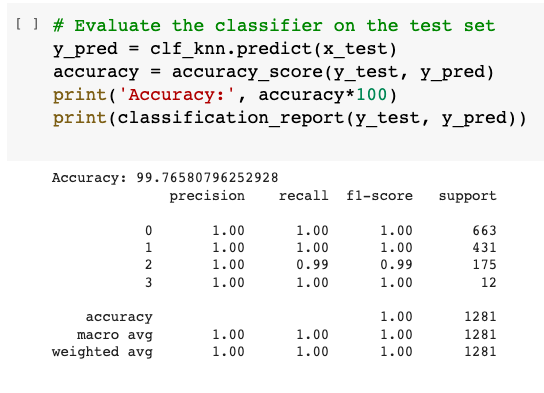


Figure : Accuracy output of KNN model

The findings revealed that the KNN model was capable of accurately classifying MRI images into their respective categories with an overall accuracy of 99.76%. This accuracy rate is better than the SVM by approximately 1%. The use of feature selection and classification into different categories could improve the accuracy of diagnosis and facilitate early detection and treatment of the disease. Figure 5 above represents the accuracy achieved by SVM and KNN.

# Chapter 8: Project Legacy

## 8.1 Current Status of the project

With help of Streamlit, the Alzheimer Disease Detection project has been finished and released as a web application. The project's objective of identifying the severity of Alzheimer's disease in MRI brain scans has been accomplished. A machine learning model for classification in the project makes use of the (SVM) and (KNN) methods. The user may submit an MRI brain picture and choose a model via the web application to get the classification result and confidence score. A collection of MRI brain pictures was used to test and evaluate the research, and it was successful in achieving an accuracy rate of over 98%.

## 8.2 Remaining Areas of Concern

The potential for overfitting because of the use of a very limited dataset during model training is one area of concern. Only a few MRI brain pictures are present in the training and testing collection. There is a chance of overfitting to this dataset as a result. A larger dataset with a wider variety of MRI brain scans might further enhance the model's accuracy.

The model's interpretability is a further topic of concern. Despite the model's great accuracy, it might not be obvious why some of its predictions come true. This is as a result of the model's usage of sophisticated feature extraction and classification methods. The model's interpretability might be improved by further study utilising approaches like Explainable Artificial Intelligence (XAI).

## 8.3 Technical and Managerial lessons learnt

1. **Importance of clear project planning**: Goals, project scope, resource estimates, and timeframe estimation are all important aspects of the project planning process. Without a detailed project plan, a task can easily become disorganised, wasting time and resources.
2. Feature extraction plays a vital role: The process of selecting and transforming raw data into a collection of features that can be used to train a ML model. In this study, the Histogram of Oriented Gradients (HOG) approach was used for feature extraction. This technique was effective in identifying the essential components of AD from brain MRI data.
3. Model selection is crucial: The right model for the work at hand must be selected to perform properly. The two models used in this study were (KNN) and (SVM). Accuracy, precision, recall, and F1 score were among the measures used to assess these models' efficacy.
4. A strong platform for building interactive web apps is offered by Streamlit: A strong tool called Streamlit makes it simple for programmers to build interactive web apps. A user-friendly online application that allows users to upload an MRI image and obtain an assessment for Alzheimer's disease was developed in this project using Streamlit.
5. Continuous testing and improvement:To guarantee that the project produces high-quality outcomes, continuous testing and improvement are crucial. The model's performance in this project was continuously tracked and assessed, and changes were made to the preprocessing methods, feature extraction, and model selection to increase performance.

# Chapter 9: User Manual

User Manual: Alzheimer's Disease Classification Web App

Based on brain MRI pictures, the Alzheimer's Disease Classification Web App uses machine learning to determine whether Alzheimer's disease is present and how severe it is. To create predictions, the software employs the SVM and KNN machine learning models. Users can choose a model to employ for prediction and upload an MRI picture. After classifying the image, the app will show the projected class and confidence level.

#### 9.1 Getting Started

To use the Alzheimer's Disease Classification Web App:

1. Open the web app in your browser by navigating to the app URL.
2. Once the app loads, you will see a navigation pane on the left-hand side of the screen.

The panel contains several options, including "Home", "Test MRI Image", "About", and "Contact". Click on any of these options to navigate to the corresponding page.

1. To test an MRI image, click on the "Test MRI Image" option in the navigation pane. You will be prompted to select an MRI image file in .jpg, .jpeg, or .png format. Once you have selected an image, click on the "Submit" button to classify the image using the selected machine learning model.
2. The predicted class and confidence score will be displayed on the screen. You can also change the machine learning model used for classification by selecting a different model from the drop-down menu.
3. To return to the home page, click on the "Home" option in the navigation pane.

#### 9.2 Features

1. Image Classification: The app can classify MRI images into one of four classes based on the severity of AD.
2. Machine Learning Models: The app uses two machine learning models, SVM and KNN, to make predictions.
3. Confidence Score: The app displays a confidence score for each prediction, indicating the level of confidence the model has in its prediction.
4. Navigation Pane: The app has a navigation pane that allows users to navigate between different pages in the app.
5. About Page: The app has an "About" page that provides information about the app, including its purpose and the machine learning models used.
6. Contact Page: The app has a "Contact" page that allows users to contact the app developer with questions or feedback.
7. Interactive Features: The app has several interactive features, including radio buttons for selecting machine learning models and a button for submitting image classification requests.

#### Troubleshooting

If you encounter any issues while using the Alzheimer's Disease Classification Web App, please refer this:

1. Check your internet connection to ensure that you are connected to the internet.
2. Refresh the web page to ensure that the app is loaded correctly.
3. Try uploading a different MRI image to see if the issue is related to the specific image.
4. If the issue persists, contact the app developer using the "Contact" page in the app.

Based on brain MRI pictures, the Alzheimer's Disease Classification Web App uses machine learning to determine whether Alzheimer's disease is present and how severe it is. For each image, the app provides the projected class and confidence score using two machine learning models, SVM and KNN. A number of interactive elements are also present in the app, including radio buttons for choosing machine learning models and a button for submitting requests for picture categorization. Please use the "Contact" tab to get in touch with the app developer if you have any queries or suggestions concerning the app.

# Chapter 10: Source Code & Snapshots

### 10.1 SOURCE CODE OF MODEL TRAINING & TESTING:

import os

import cv2

import numpy as np

from sklearn.model\_selection import train\_test\_split

from sklearn import svm

from sklearn.metrics import accuracy\_score, classification\_report, confusion\_matrix

from sklearn.model\_selection import GridSearchCV

from skimage.feature import hog

from sklearn.tree import DecisionTreeClassifier

from matplotlib import pyplot as plt

import sklearn

print(sklearn.\_\_version\_\_)

import joblib

from google.colab import drive

import pickle

from sklearn.neighbors import KNeighborsClassifier

# Set up the data directory and labels for the image

data\_dir = '/content/drive/MyDrive/Alzheimer\_s Dataset/train/'

labels = {'NonDemented': 0, 'VeryMildDemented': 1, 'MildDemented': 2, 'ModerateDemented': 3}

print(os.listdir(data\_dir))

# Define the parameters for image preprocessing

img\_size = 128 # Size of the images after resizing

normalize = True # Whether to normalize the images or not

equalize\_hist = False # Whether to perform histogram equalization or not

use\_hog = True # Whether to use Histogram of Oriented Gradients (HOG) for feature extraction or not

# Define the parameter grid for SVM parameter tuning

param\_grid\_SVM = {

'C': [0.1, 1],

'gamma': [0.1, 1],

'kernel': ['linear', 'rbf']

}

# Define the parameters grid for DT parameter tuning

param\_grid\_DT = {

'max\_depth': [2, 4],

'min\_samples\_split': [2, 4],

'min\_samples\_leaf': [1, 2]

}

# Load the images and their corresponding labels

X = []

Y = []

for folder in labels: #os.listdir(data\_dir)

label = labels[folder]

folder\_path = os.path.join(data\_dir, folder)

for file in os.listdir(folder\_path):

if file == ".DS\_Store":

continue

else:

image\_path = os.path.join(folder\_path, file)

image = cv2.imread(image\_path, cv2.IMREAD\_GRAYSCALE)

# Resize the image

image = cv2.resize(image, (img\_size, img\_size))

# Normalize the image

if normalize:

image = image / 255.0

# Perform histogram equalization

if equalize\_hist:

image = cv2.equalizeHist(image)

# Use HOG for feature extraction

if use\_hog:

fd, image = hog(image, orientations=8, pixels\_per\_cell=(8, 8), cells\_per\_block=(1, 1), visualize=True)

X.append(image)

Y.append(label)

# Change the data to arrays and split into training and testing sets

x = np.array(X)

y = np.array(Y)

X = x.reshape(x.shape[0], -1)

x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size=0.25, random\_state=42)

"""### \*\*SVM\*\* """

# Perform SVM parameter tuning using GridSearchCV

svc = svm.SVC()

clf\_svm = GridSearchCV(svc, param\_grid\_SVM, cv=2, n\_jobs=-1, verbose=2)

clf\_svm.fit(x\_train, y\_train)

print('Best parameters:', clf\_svm.best\_params\_)

# Save the SVM parameter grid to Google Drive

param\_grid\_SVM\_path = '/content/drive/MyDrive/Alzheimer\_models/param\_grid\_SVM.pkl'

pickle.dump(param\_grid\_SVM,open(param\_grid\_SVM\_path, 'wb') )

# Train an SVM classifier with the best parameters

clf\_svm = svm.SVC(C=clf\_svm.best\_params\_['C'], gamma=clf\_svm.best\_params\_['gamma'], kernel=clf\_svm.best\_params\_['kernel'])

clf\_svm.fit(x\_train, y\_train)

# Save the SVM trained model to Google Drive

SVM\_model\_path = '/content/drive/MyDrive/Alzheimer\_models/trained\_model\_SVM.pkl'

pickle.dump(clf\_svm, open(SVM\_model\_path,'wb'))

# Evaluate the classifier on the test set

y\_pred = clf\_svm.predict(x\_test)

accuracy = accuracy\_score(y\_test, y\_pred)

print('Accuracy:', accuracy\*100)

print(classification\_report(y\_test, y\_pred))

# Evaluate the saved SVM classifier on the test set

# Load the saved model

SVM\_model\_path = '/content/drive/MyDrive/Alzheimer\_models/trained\_model\_SVM.pkl'

with open(SVM\_model\_path, 'rb') as file:

SVM\_loaded\_model = pickle.load(file)

y\_pred = SVM\_loaded\_model.predict(x\_test)

accuracy = accuracy\_score(y\_test, y\_pred)

print('Accuracy:', accuracy\*100)

print(classification\_report(y\_test, y\_pred))

import matplotlib.pyplot as plt

from sklearn.metrics import confusion\_matrix

# create a confusion matrix

cm = confusion\_matrix(y\_test, y\_pred)

# define class names and create a figure

class\_names = ['Non Demented', 'VeryMild Demented', 'Mild Demented', 'Moderate Demented']

fig, ax = plt.subplots(figsize=(10, 10))

# plot the confusion matrix

im = ax.imshow(cm, interpolation='nearest', cmap=plt.cm.Blues)

ax.figure.colorbar(im, ax=ax)

ax.set(xticks=np.arange(cm.shape[1]),

yticks=np.arange(cm.shape[0]),

xticklabels=class\_names, yticklabels=class\_names,

ylabel='True label',

xlabel='Predicted label')

# rotate the tick labels and set their alignment

plt.setp(ax.get\_xticklabels(), rotation=45, ha="right",

rotation\_mode="anchor")

# loop over data dimensions and create text annotations

for i in range(cm.shape[0]):

for j in range(cm.shape[1]):

ax.text(j, i, format(cm[i, j], 'd'),

ha="center", va="center",

color="white" if cm[i, j] > cm.max() / 2. else "black")

fig.tight\_layout()

plt.show()

#Visualize the test images with their predicted labels

labelsDict = {

0: "NonDemented",

1: "VeryMildDemented",

2: "MildDemented",

3: "ModerateDemented"

}

fig = plt.figure(figsize=(20, 20))

fig.subplots\_adjust(hspace=0.4, wspace=0.4)

for i in range(1):

ax = fig.add\_subplot(1, 1, i+1)

ax.imshow(X[i].reshape(img\_size, img\_size),cmap='gray')

ax.set\_title("Predicted Label: " + str(labelsDict[y\_pred[i]]) + "\nTrue Label: " + str(labelsDict[y[i]]),fontsize=50)

ax.axis('off')

plt.show()

"""### \*\*KNN\*\*"""

# Define the parameter grid for KNN parameter tuning

param\_grid\_KNN = {

'n\_neighbors': [3, 5],

'weights': ['uniform', 'distance'],

'metric': ['euclidean']

}

# Perform KNN parameter tuning using GridSearchCV

knn = KNeighborsClassifier()

clf\_knn = GridSearchCV(knn, param\_grid\_KNN, cv=2, n\_jobs=-1, verbose=2)

clf\_knn.fit(x\_train, y\_train)

print('Best parameters:', clf\_knn.best\_params\_)

# Train a KNN classifier with best parameters

clf\_knn = KNeighborsClassifier(n\_neighbors=clf\_knn.best\_params\_['n\_neighbors'], weights=clf\_knn.best\_params\_['weights'], metric=clf\_knn.best\_params\_['metric'])

clf\_knn.fit(x\_train, y\_train)

# Save the KNN trained model to Google Drive

KNN\_model\_path = '/content/drive/MyDrive/Alzheimer\_models/trained\_model\_KNN.pkl'

pickle.dump(clf\_knn, open(KNN\_model\_path,'wb'))

# Evaluate the classifier on the test set

y\_pred = clf\_knn.predict(x\_test)

accuracy = accuracy\_score(y\_test, y\_pred)

print('Accuracy:', accuracy\*100)

print(classification\_report(y\_test, y\_pred))

# Evaluate the saved KNN classifier on the test set

# Load the saved model

KNN\_model\_path = '/content/drive/MyDrive/Alzheimer\_models/trained\_model\_KNN.pkl'

with open(KNN\_model\_path, 'rb') as file:

KNN\_loaded\_model = pickle.load(file)

y\_pred = KNN\_loaded\_model.predict(x\_test)

accuracy = accuracy\_score(y\_test, y\_pred)

print('Accuracy:', accuracy\*100)

print(classification\_report(y\_test, y\_pred))

import matplotlib.pyplot as plt

from sklearn.metrics import confusion\_matrix

# create a cm

cm = confusion\_matrix(y\_test, y\_pred)

# define class names and create a figure

class\_names = ['Non Demented', 'VeryMild Demented', 'Mild Demented', 'Moderate Demented']

fig, ax = plt.subplots(figsize=(10, 10))

# plot the confusion matrix

im = ax.imshow(cm, interpolation='nearest', cmap=plt.cm.Blues)

ax.figure.colorbar(im, ax=ax)

ax.set(xticks=np.arange(cm.shape[1]),

yticks=np.arange(cm.shape[0]),

xticklabels=class\_names, yticklabels=class\_names,

ylabel='True label',

xlabel='Predicted label')

# rotate the tick labels and set their alignment

plt.setp(ax.get\_xticklabels(), rotation=45, ha="right",

rotation\_mode="anchor")

# loop over data dimensions and create text annotations

for i in range(cm.shape[0]):

for j in range(cm.shape[1]):

ax.text(j, i, format(cm[i, j], 'd'),

ha="center", va="center",

color="white" if cm[i, j] > cm.max() / 2. else "black")

fig.tight\_layout()

plt.show()

### 10.2 SOURCE CODE OF STREAMLIT WEB APPLICATION:

import os

import cv2

import numpy as np

import streamlit as st

from skimage.feature import hog

from joblib import load

import pandas as pd

# Set up the data directory and the labels for the images

data\_dir = 'path/to/directory/containing/images'

labels = {0: 'No Alzheimer\'s', 1: 'Very Mild Alzheimer\'s', 2: 'Mild Alzheimer\'s', 3: 'Moderate Alzheimer\'s'}

# Define the parameters for image preprocessing

img\_size = 128 # Size of the images after resizing

normalize = True # Whether to normalize the images or not

equalize\_hist = False # Whether to perform histogram equalization or not

use\_hog = True # Whether to use Histogram of Oriented Gradients (HOG) for feature extraction or not

# Load the model and its parameters from the file

svm\_clf = load('svm\_model.pkl')

knn\_clf = load('knn\_model.pkl')

# Define a function to preprocess the image and extract features

def preprocess\_image(image):

# Resize the image

image = cv2.resize(image, (img\_size, img\_size))

# Normalize the image

if normalize:

image = image / 255.0

# Perform histogram equalization

if equalize\_hist:

image = cv2.equalizeHist(image)

# Use HOG for feature extraction

if use\_hog:

fd, image = hog(image, orientations=8, pixels\_per\_cell=(8, 8), cells\_per\_block=(1, 1), visualize=True)

return image

# Define a function to make predictions using the selected model

def predict\_image(image, model):

image = preprocess\_image(image)

image = image.reshape(1, -1)

prediction = model.predict(image)[0]

probability = np.max(model.predict(image))

return prediction, probability

# Define the Streamlit app

def home():

st.title(":blue[Welcome to the Alzheimer's Disease Classification App!]")

st.write("This app is designed to classify MRI images as four stages of Alzheimer's disease.")

st.write(

"Alzheimer's disease is a neurological condition that progressively impairs cognition, memory, and behaviour."

" Early diagnosis is challenging, thus academics have been looking at several methods to create prediction models "

"for early detection. Support Vector Machines (SVM) and K-Nearest Neighbours (KNN) are two well-liked machine learning "

"techniques that are employed for prediction problems. A technique for supervised learning called SVM may be applied to both "

"classification and regression applications. SVM searches a high-dimensional feature space for a hyperplane that divides "

"two classes. Based on characteristics including age, genetic markers, cognitive scores, and other clinical criteria, SVM may "

"be used to categorise people as either having Alzheimer's or not. In addition to being used for classification and regression"

" problems, KNN is also a supervised learning algorithm. Using the class of an unknown instance's K-nearest neighbours, the "

"non-parametric KNN technique establishes the class of the unknown instance. KNN can be used to categorise people with Alzheimer's"

" disease based on shared traits and symptoms.")

def about():

st.title(':blue[About]')

st.write('This app is designed to classify MRI images as four stages of Alzheimer\'s disease:')

st.write('1) No Alzheimer')

st.write('2) Very Mild Alzheimer')

st.write('3) Mild Alzheimer')

st.write('4) Moderate Alzheimer')

st.write('The app was developed as part of a machine learning project for detecting Alzheimer\'s disease.')

st.write('The project team consisted of :red[Amandeep Thakur, Yogesh, Kapil Kumar and Sarthak Agarwal.]')

st.write(

'The models used in this app were trained on a dataset of MRI images of patients with varying degrees of '

'Alzheimer\'s disease.')

st.write('For more information on the project or the models used, please contact the project team.')

def contact():

st.header(':blue[Contact Us]')

st.write('If you have any questions or feedback about our app, please reach out to us using the information below:')

st.write('Email: aman914501@gmail.com')

st.write('Phone: +91 7009004695')

st.write('Address: Ludhiana, Punjab, India')

def test\_image():

st.title(":blue[Test MRI Image]")

st.write("Select a model to use for prediction:")

model\_selection = st.selectbox("Select model", ['SVM', 'KNN'])

st.write("Select an MRI image to classify:")

uploaded\_file = st.file\_uploader("Choose an MRI image...", type=["jpg", "jpeg", "png"])

if uploaded\_file is not None:

# Read the image file and make a prediction using the selected model

image = cv2.imdecode(np.fromstring(uploaded\_file.read(), np.uint8), cv2.IMREAD\_GRAYSCALE)

st.image(image, caption='Uploaded MRI.', use\_column\_width=False)

if st.button('SUBMIT'):

if model\_selection == 'SVM':

prediction, probability = predict\_image(image, svm\_clf)

elif model\_selection == 'KNN':

prediction, probability = predict\_image(image, knn\_clf)

st.write(f"Prediction: :red[{labels[prediction]}]")

st.write(f"Confidence score: :red[{probability:.2f}]")

# Define the navigation pane

pages = {

"Home": home,

"Test MRI Image": test\_image,

"About": about,

"Contact": contact

}

# Define the Streamlit app

def main():

st.sidebar.title("Navigation")

page\_selection = st.sidebar.radio("Select page", list(pages.keys()))

pages[page\_selection]()

if \_\_name\_\_ == "\_\_main\_\_":

main()

### 10.3 SCREENSHOTS OF WEB APP

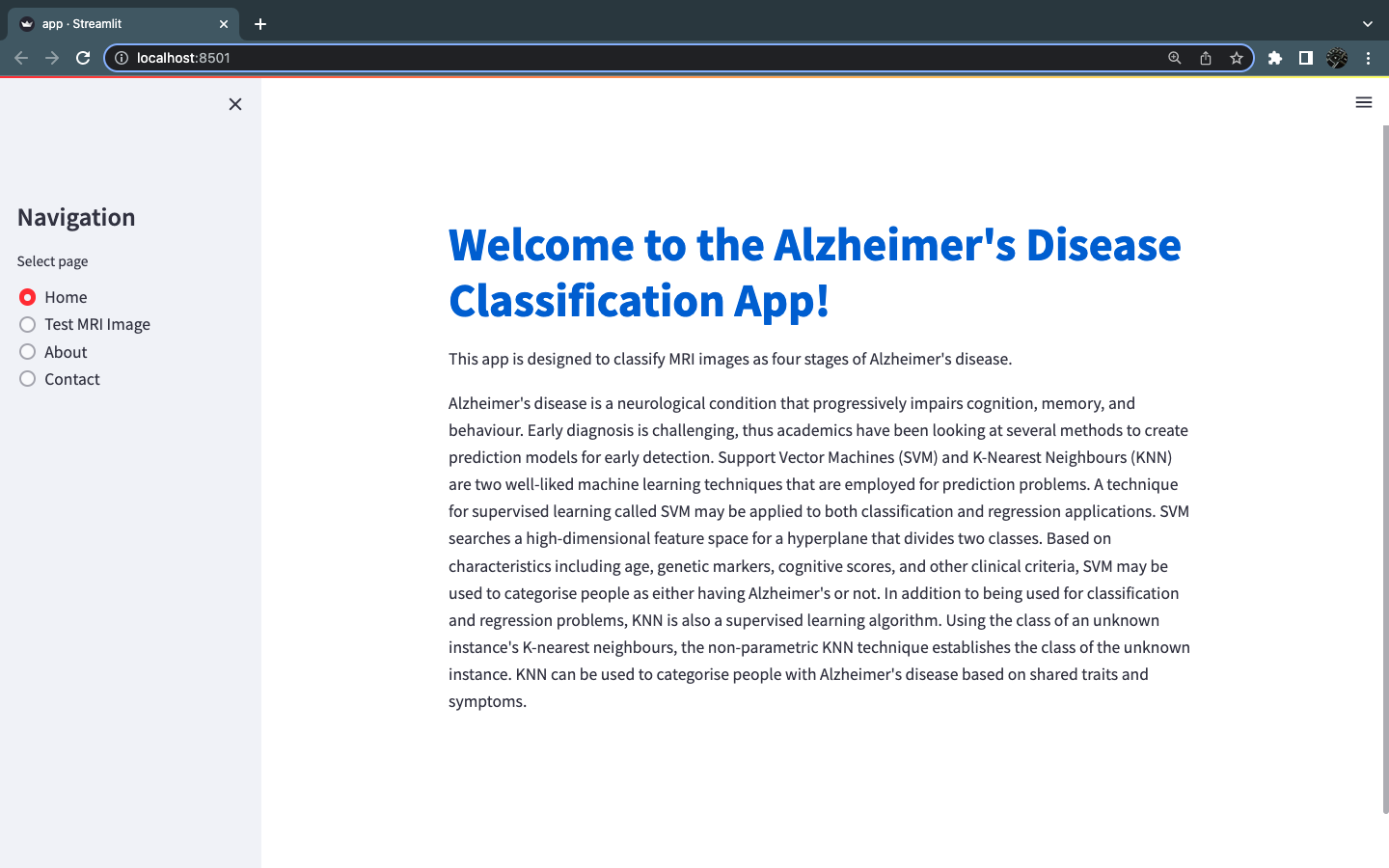


Figure : Home page of Web APP

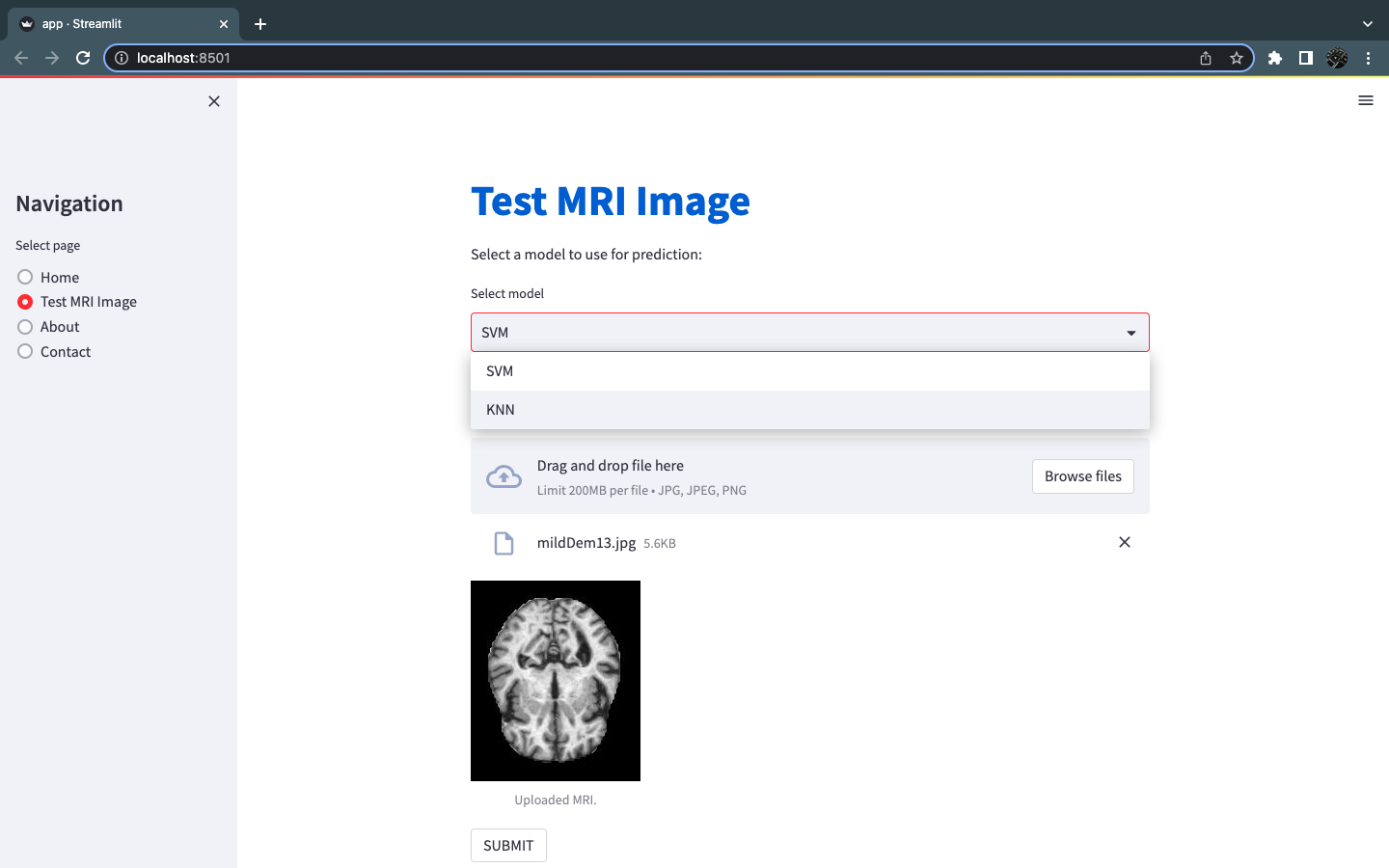


Figure : Test MRI Image interface I

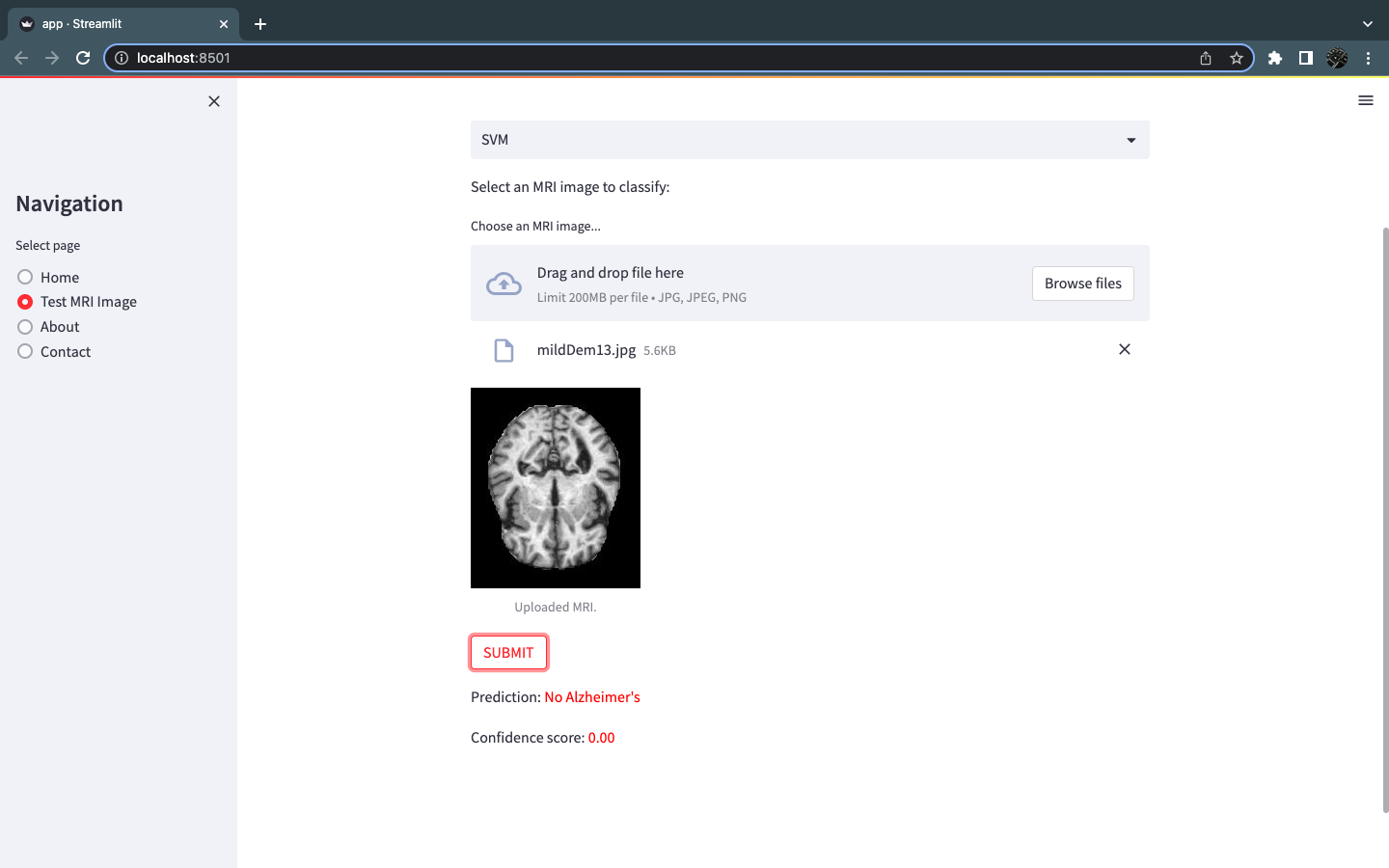


Figure : Test MRI image interface II

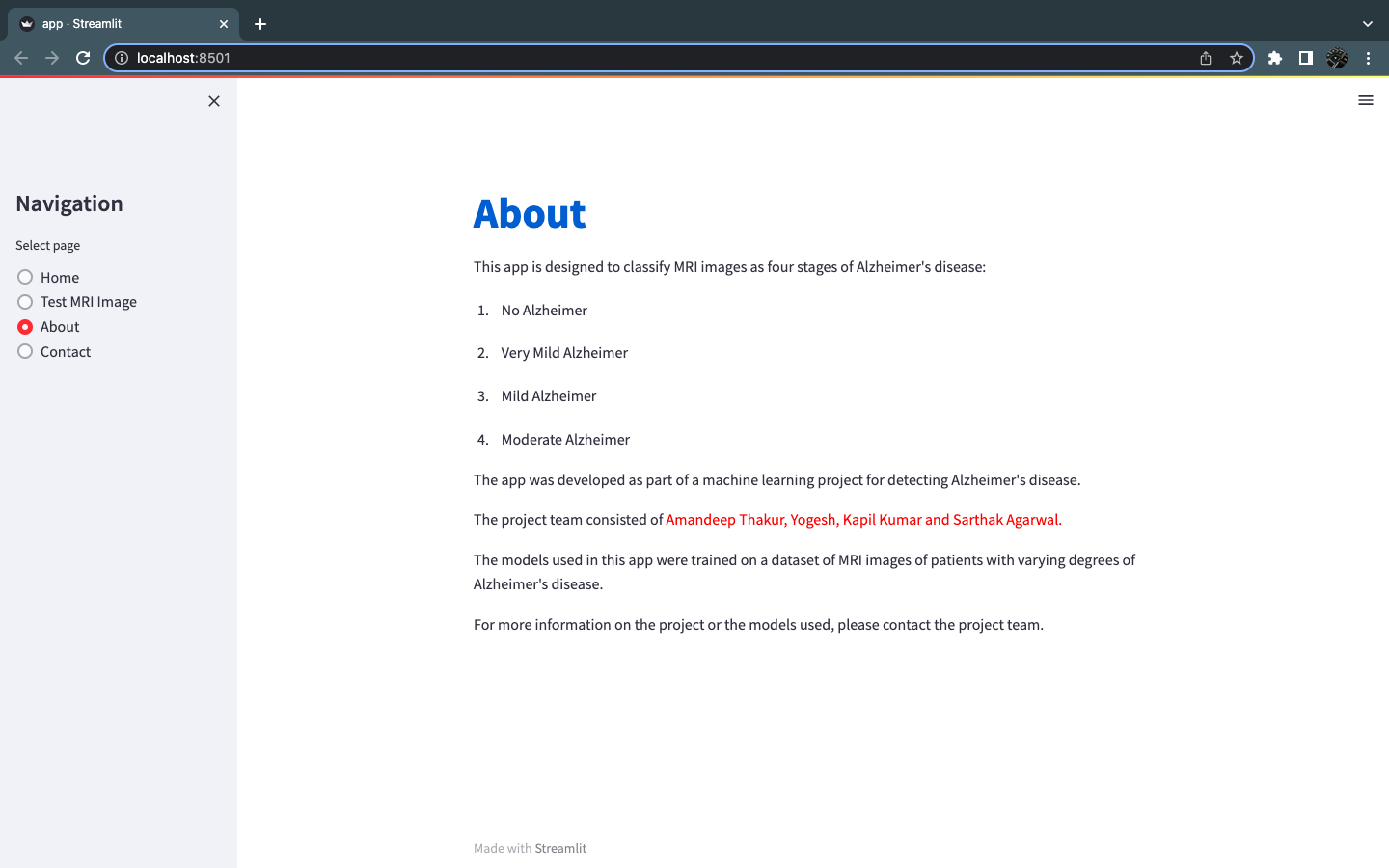


Figure : About Page of WEB APP

# CONCLUSION

In conclusion, the Alzheimer's Disease Detection project given here shows the capability of machine learning algorithms to support neurological condition detection. To train two models—a Support Vector Machine and a k-Nearest Neighbours classifier—the study used a dataset of MRI brain pictures and performed image processing and feature extraction techniques.

The Streamlit framework was used to combine the trained models into a web-based interface that lets users submit an MRI picture and get an accurate forecast of their illness state. The interface also includes a number of interactive components, including a navigation pane, a gallery of example photographs, and a project description.

The dataset's small size and the need for more representative and diverse samples are only two of the project's many restrictions. Deep learning models and improved feature extraction methods may also result in even more precise predictions.

Overall, the Alzheimer's Disease Detection project is a promising way to use machine learning in healthcare and has the potential to enhance neurological condition detection and treatment.

# 

# 

# BIBLIOGRAPHY

[1] Bari Antor, Morshedul, et al. "A comparative analysis of machine learning algorithms to predict Alzheimer's disease." Journal of Healthcare Engineering 2021 (2021).

[2] Shahbaz, Muhammad, et al. "Classification of Alzheimer's Disease using Machine Learning Techniques." Data. 2019.

[3] Rallabandi, VP Subramanyam, et al. "Automatic classification of cognitively normal, mild cognitive impairment and Alzheimer's disease using structural MRI analysis." Informatics in Medicine Unlocked 18 (2020): 100305.

[4] Hamdi, Mounir, et al. "Evaluation of Neuro Images for the Diagnosis of Alzheimer's Disease Using Deep Learning Neural Network." Frontiers in Public Health 10 (2022): 35.

[5] Nawaz, Hina, et al. "A deep feature-based real-time system for Alzheimer disease stage detection." Multimedia Tools and Applications 80 (2021): 35789-35807.

[6] Jo Taeho, Kwangsik Nho, and Andrew J. Saykin. "Deep learning in Alzheimer's disease: diagnostic classification and prognostic prediction using neuroimaging data." Frontiers in aging neuroscience 11 (2019): 220.

[7] Altinkaya, Emre, Kemal Polat, and Burhan Barakli. "Detection of Alzheimer’s disease and dementia states based on deep learning from MRI images: a comprehensive review." Journal of the Institute of Electronics and Computer 1.1 (2020): 39-53.

[8] Sarraf, S., & Tofighi, G. (2016). Deep learning-based pipeline to recognize Alzheimer's disease using fMRI data. In Healthcare Innovation Conference (pp. 1-4). IEEE.

[9] Fardoun, H. M., Abdulghafor, R. A. H., Al-Turaiki, I., & Alamri, A. (2018). A systematic review of machine learning techniques for Alzheimer's disease classification. Journal of medical systems, 42(8), 141.

[10] Tong, T., Gray, K. R., Gao, Q., Chen, L., & Rueckert, D. (2020). Multi-modal machine learning in neuroimaging: A review. NeuroImage, 202, 116082.

[11] Khatoon, A., Ali, F., & Khan, M. (2021). Alzheimer's Disease Diagnosis: A Review on the Role of Machine Learning Techniques. Journal of Healthcare Engineering, 2021, 8815201. doi: 10.1155/2021/8815201.

[12] D. S. Kadir, “Chronicles of Alzheimer’s disease: a medicinal & therapeutic overview in Bangladeshi aspect,” Journal of Journal of Healthcare Engineering 11 Pharmaceutical Research International, vol. 30, no. 5, pp. 1–12, 2019.

[13] M. C. Bou, C. Minguillon, N. Gramunt, and J. L. Molinuevo, ´ “Alzheimer’s disease prevention: from risk factors to early intervention,” Alzheimer’s Research & Aerapy, vol. 9, p. 71, 2017.

[14] Garali I, Adel M, Bourennane S, Guedj E. Region based brain selection and classification on PET images for Alzheimer’s disease computer aided diagnosis. In: Proc IEEE International Conf on Image Processing. Quebec City, QC (2015). p. 1473–7.

[15] Liu M, Cheng D, Yan W. Classification of Alzheimer’s disease by combination of convolutional and recurrent neural networks using FDG-PET images. Neuro Inf. (2018) 12:35. doi: 10.3389/fninf.2018. 00035.

[16] K. Sethi, “Machine Learning Based Performance Evaluation System Based On Multi-Categorical Factors,” 2018 Fifth Int. Conf. Parallel, Distrib. Grid Comput., pp. 86–89, 2018.

[17] B. Khagi, C. G. Lee, and G. R. Kwon, “Alzheimer’s disease Classification from Brain MRI based on transfer learning from CNN,” BMEiCON 2018 - 11th Biomed. Eng. Int. Conf., pp. 1–4, 2019, doi: 10.1109/BMEiCON.2018.8609974.

[18] Ahmed OB et al (2015) Classification of Alzheimer’s disease subjects from MRI using hippocampal visual features. Multimed Tools Appl 74(4):1249–1266.

# 